Python – ESP32 (Aula 2)

Instrumentação Eletrotecnia Aplicada

IEA 2021-2022

A Borges

Conceitos gerais

- Python
- Python instalação

- IDE : Integrated Development Environment

(Ambiente de Desenvolvimento Integrado)

Python

- Linguagem generalista, possível de utilizar num vasto conjunto de disciplinas: biologia, química, finanças, analise numérica, robótica, etc;
- www.python.org

- Linguagens de alto nível (C, C++, Visual Basic, etc)
- Linguagens interpretadas (Python, Matlab, etc)
- Linguagens gráficas (LabVIEW)

Thonny - Instalação

1. Instalação Python (Windows)

https://www.python.org/downloads/release/python-3102/

2. Instalação do IDE Pythom-ESP32

https://thonny.org

MicroPython (Thonny) - Instalação

3. Instalar o **Drive do ESP32**

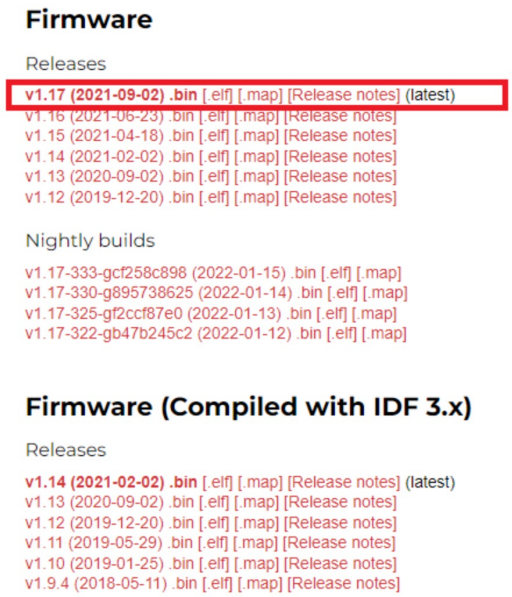
https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers

CP210x Windows Drivers

Confirmar no **Gestor de Dispositivos** se está corretamente instalado e associado a uma porta série (**COM3**... Por exemplo)

4. Dowload **Firmware (ESP32)**

https://micropython.org/download/esp32/



5. Instalação plug-ins

Tools -> Manage plug-ins -> esptool

https://pypi.org/project/esptool/

na linha de comandos do Windows, instalar o Esptools

pip install esptool

6. Selecionar **MicroPython(ESP32)**

Tools -> Interpreter -> MicroPython(ESP32)

Selecionar a porta e eventualmente realizar o upload do firmware

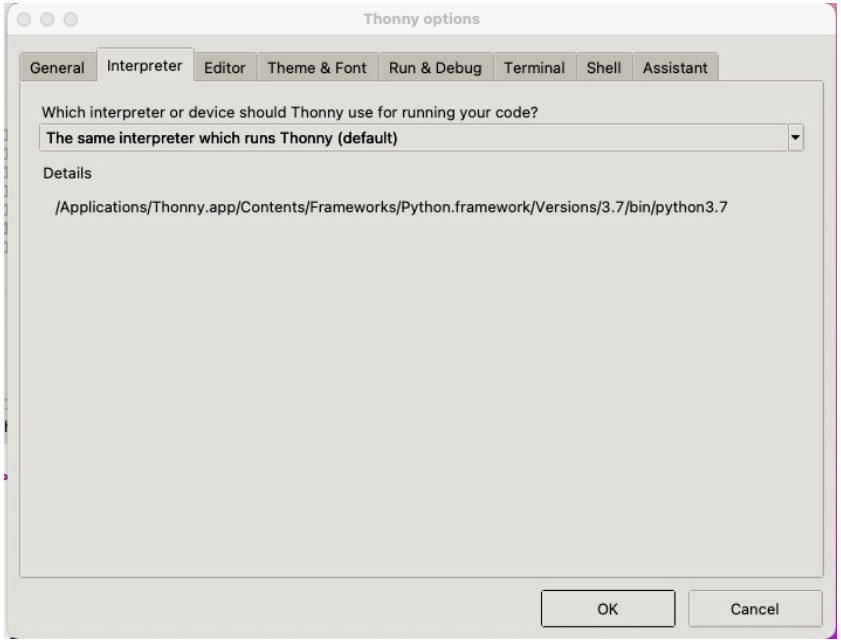
Python – ESP32 (Aula 2)

Programação em PC

IEA 2021-2022

A Borges

Python: Configuração



Preferências (utilização do Python em PC)

Funções

Exemplo 1

Teste.py | main.py * | main1.py | ConvTemp_C-FK.py * | ConvTemp_C-F2.py *

```
1  # IEA 2021-2022
2  #4 Março 2022
3  def ListaCompras():
4      print('Arroz')
5      print("açucar")
6      print ('fim rotina')
7      print ('\n')
8
9  # Programa principal
10 ListaCompras ()
11 print ('Hello World')
12
13
14
```

Shell

```
>>> %Run Teste.py

  Arroz
  açucar
  fim rotina


  Hello World

>>>
```

```
def <nome da função> (<parâmetros>):
    print ( 'Função' )          # Corpo da função
    return <valor>
    # ...

# Programa principal
<nome da função>
```

Exercício 2:

Utilizando o exemplo de conversão de temperatura ºC em ºF ou K.

Criar 3 funções:

- **LerC() -> Leitura temperatura em ºC**
- **ConvCF(Celsius) -> Conversão ºC em ºF**
- **ConvCF(Celsius) -> Conversão ºC em K**

MicroPyhon (Programming Basics) : Instruções de repetição ciclos **For**

Exemplo 2

FileReadWrite.py | For_ciclo.py | ConvTemp_C-F2.py

```
1  # Ciclo de repetição For
2  # 6 Março 2022
3
4  lista = ["bananas", "laranjas", "maça"]
5
6  for frutas in lista:
7      # visualização das frutas
8      print ( frutas )
9
10
11 # fim de programa
```

Shell

```
>>> %Run For_ciclo.py

  bananas
  laranjas
  maça

>>>
```

Exemplo 3

FileReadWrite.py | For_ciclo2.py | ConvTemp_C-F2.py

```
1  # Ciclo de repetição For (2)
2  # 6 Março 2022
3
4  print ('ciclo simples')
5  for aux in range(5):
6      # visualização numérico
7      print ( aux )
8
9  print ('ciclo completo')
10 for aux in range(1,10,3):
11     # visualização numérico
12     print ( aux )
13
14 # fim de programa
```

Shell

```
>>> %Run For_ciclo2.py

  ciclo simples
  0
  1
  2
  3
  4
  ciclo completo
  1
  4
  7

>>>
```

for item in <sequencia>:
**   # Instruções de repetição**
**   <código..>**

for aux in range (inicio, fim, incremento):
**   # Instruções de repetição**
**   <código..>**

MicroPyhon (Programming Basics) : funções de temporização

Exemplo 5

```
from time import sleep


for i in range(10):
    sleep (0.5)
    print (i)
```

from time import sleep

for i in range(10):
** sleep (0.5)**
** print (i)**

Exercício 6 : PISCA PISCA

Utilizar um **ciclo while**, e excute um pisca pisca de 10 ciclos, indicando a **mensagem "ON" – "OFF"**

Realize um **pisca pisca:**

0.5 s	**ON**
1 s	**OFF**

Leitura data e hora (Python 3.7.9)

```
# import module read date and time
from datetime import datetime

# Read date and time
now = datetime.now()

# Format date and time
current_time = now.strftime("%Y-%m-%d  %H:%M:%S")
print ("Current time = ", current_time )
print ("******************" )
print ("Now : ", now )
```

%b – nome abreviado do mês
%d - dia
%H – Hora [0..23]
%I – Hora [01..12]
%m – Mês
%M – Minutos
%p – AM or PM
%S – Segundos
%y – Ano [00..99]
%Y – Ano

Exemplo 7

```
# Leitura da Data-Hora Python
# 12 Março 2022

from datetime import datetime

now = datetime.now()
current_time = now.strftime("%Y-%M-%d  %H:%M:%S")
print ( current_time )
```

Leitura data e hora (MicroPython)

import utime as time

Read date and time

aux = **time.gmtime()**

print (aux)

format date and time

name = '{:04d}-{:02d}-{:02d} {:02d}:{:02d}:{:02d} '.format(aux[0],aux[1],aux[2],aux[3],aux[4],aux[5],)

print (name)

[Resultado]

>>> print (aux)

(2022, 3, 12, 20, 20, 54, 5, 71)

Exemplo 8

```
# Leitura da Data-Hora MicroPython
# 12 Março 2022

import utime as time

now = time.gmtime()
ff = '{:04d}-{:02d}-{:02d}  {:02d}:{:02d}:{:02d}'.format(now[0],now[1],now[2],now[3],now[4],now[5])
print ( ff )
```

MicroPyhon (Programming Basics) : Gestão Ficheiros

Exemplo 9

```
# Escrita e Leitura em Ficheiro – Python – MicroPython
# 13 Março 2022

# Abertura ficheiro
f = open ("Demo2.txt","w")
f.write ('Escreve uma linha em ficheiro de texto... \n')
f.write ('escreve outra linha em ficheiro de texto... \n')
f.close


# Visualiza o que escreveu em ficheiro de texto
f = open ("Demo2.txt","r")
xx = f.read()
print ( xx )
f.close
```

f = open (“nome do ficheiro.txt”, <tipo acesso>)

f.write (<valores ou texto>)

f.close

f.read()

<tipo de acesso>:

“x” – create, retorna um erro caso exista

“a” – Append

“w” – Escrita, cria ficheiro novo

“r” - Leitura

‘\n’ muda de linha

{:02d} - formatação 2 casas décimais

Exercício:

Exercício 10

Escrever em ficheiro com 10 linhas para registo dos alunos da turma prática.

Dados a registar:

- Registo do numero de entrada,
- Data – Hora
- Num Mecanográfico
- Nome (1º e último)

Após o registo, o ficheiro deve ser visualizado.

```
>>> %Run 'Aula2_RegistoAlunosTP 2022_03_13.py'

  Qual a turma prática : 1

  *****
  Registo dos alunos:
  0
  Número mecanográfico : 1
  Nome do aluno : aluno 1
  1
  Número mecanográfico : 2
  Nome do aluno : aluno 2
  2
  Número mecanográfico : 3
  Nome do aluno : aluno 3
  3
  Número mecanográfico : 4
  Nome do aluno : aluno 4
  4
  Número mecanográfico : 5
  Nome do aluno : aluno 5

  ********************
  Visualização do registo dos alunos...

  Registo dos alunos da turma prática : 1


   Registo N : 00  2022-03-13  11:34:35  1 - aluno 1
   Registo N : 01  2022-03-13  11:34:40  2 - aluno 2
   Registo N : 02  2022-03-13  11:34:44  3 - aluno 3
   Registo N : 03  2022-03-13  11:34:47  4 - aluno 4
   Registo N : 04  2022-03-13  11:34:51  5 - aluno 5
  Fim registo
```

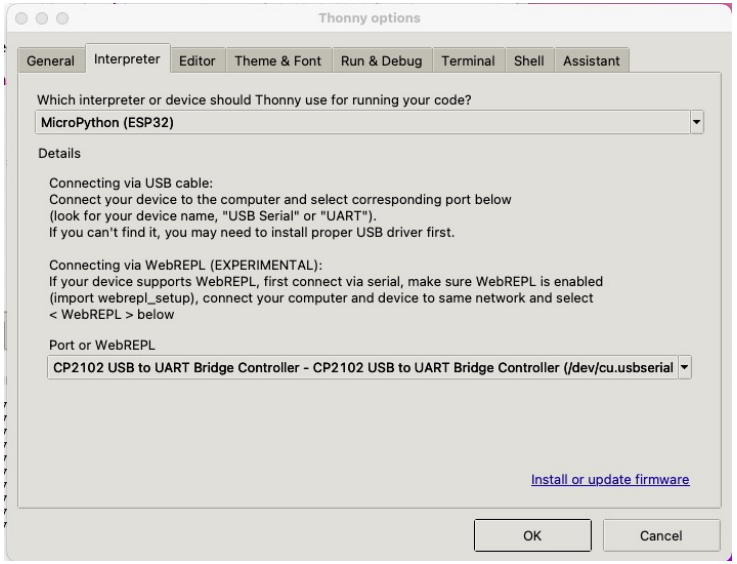
Python – ESP32

ESP32 - Programas

IEA 2021-2022

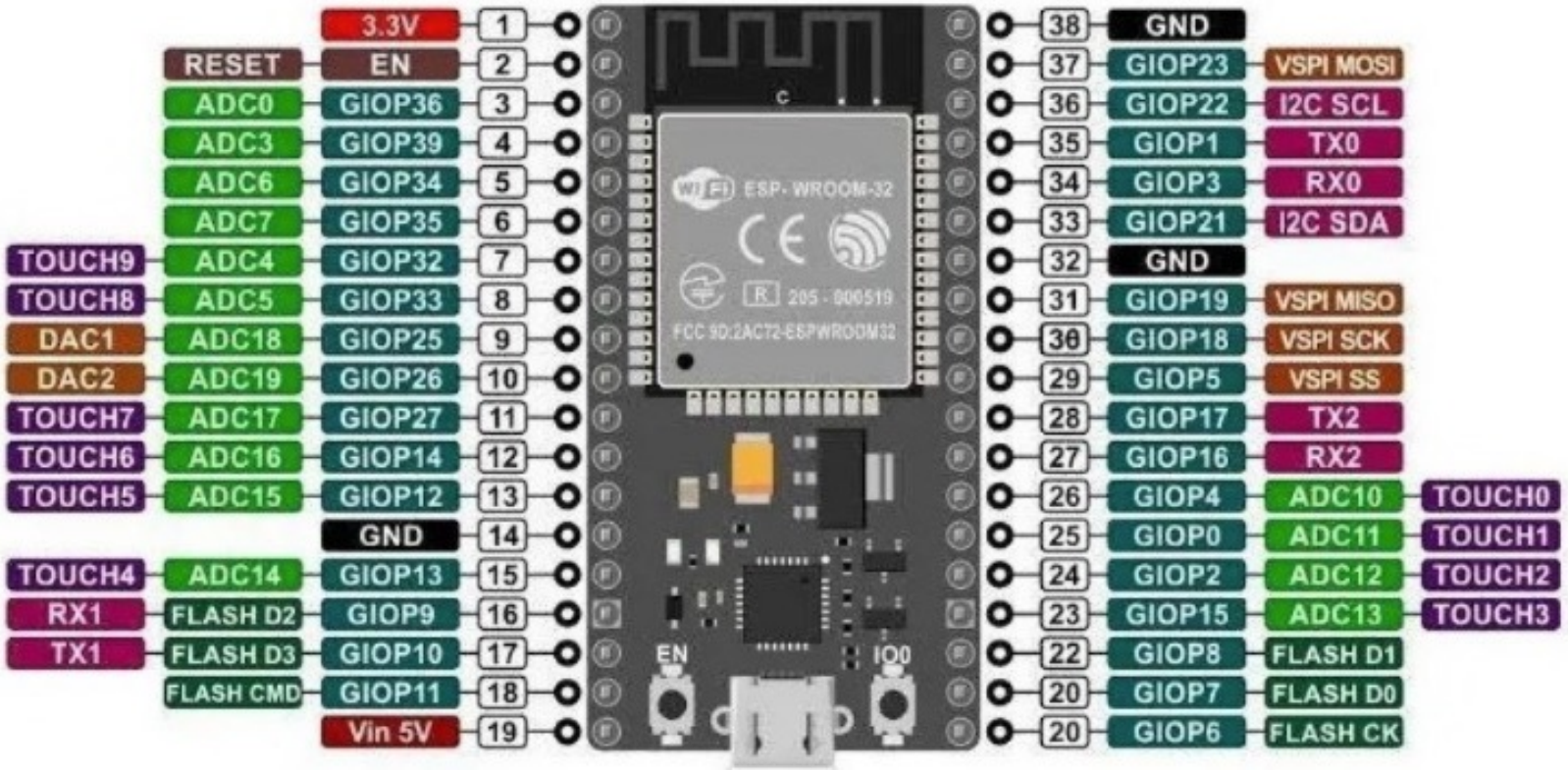
A Borges

MicroPyhon (ESP32):

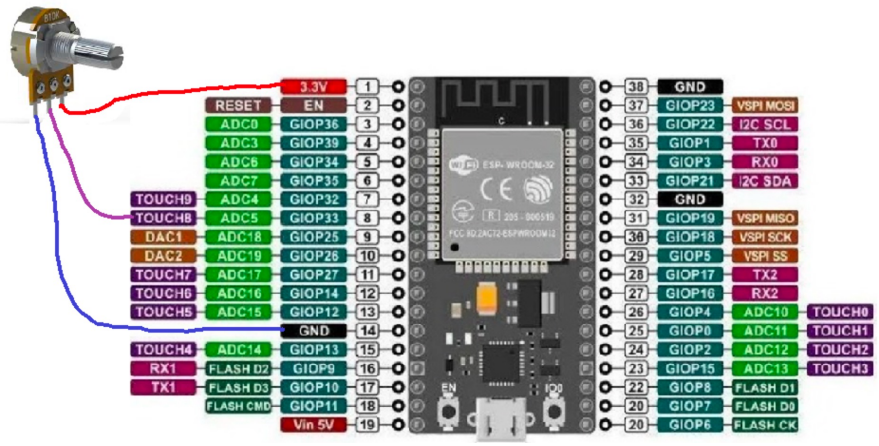


Preferências

MicroPyhon (ESP32) : Diagrama



MicroPyhon Leitura Entrada Analógica



- **ADC – Analogic Digital Converter**

- **Conversor analógico digital 0 a 3.3 V**
- **12 Bits – 2^{12} – 4096 pontos**
- **Corresponde a 3.3 V / 4096 = 0,000805 V**

Programa para leitura entrada analógica:

```
from machine import Pin, ADC

pot = ADC(Pin(33))              # criação do objeto
pot.atten (ADC.ATTN_11DB)       # full range 3.3V

xx = pot.read()
print ('Valor ADC : ', xx   )
```

Exercício 11:

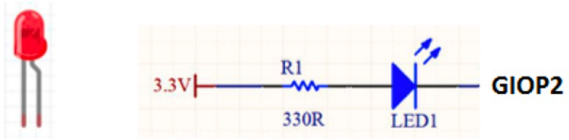
Ciclo com 100 leituras da entrada analógica, visualização do valor numérico e do valor convertido para tensão.

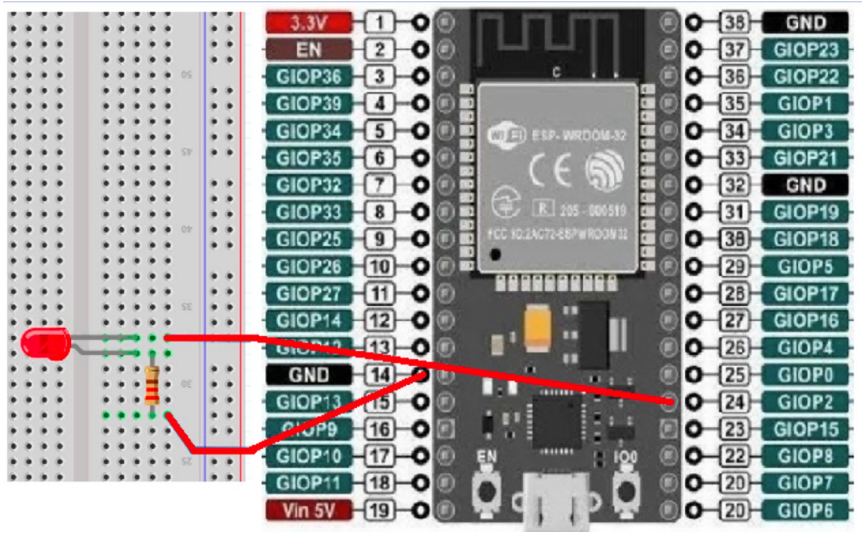
MicroPyhon (Programming Basics) : Pisca Pisca

Exemplo 12

```
# Led Pisca Pisca 0.5 s
# 13 Março 2022

from time import sleep
from machine import Pin

# Configração da entrada
led = Pin ( 2, Pin.OUT)
led.value (True)
aux = True

xx = input ('Inicio : ' )

while True:
    led.value ( not led.value() )
    aux = not aux
    sleep (0.5)
    print ( 'Pisca Pisca : ', aux )
```

```
>>> %Run -c $EDITOR_CONTENT

  Inicio : 1
  Pisca Pisca :  False
  Pisca Pisca :  True
  Pisca Pisca :  False
  Pisca Pisca :  True
  Pisca Pisca :  False
  Pisca Pisca :  True
  Pisca Pisca :  False
  Pisca Pisca :  True
  Pisca Pisca :  False
```





Notas:

Pisca de 0,5 segundos – ciclo infinito

Python – ESP32

Notas Várias

IEA 2021-2022

A Borges

Sistemas de numeração

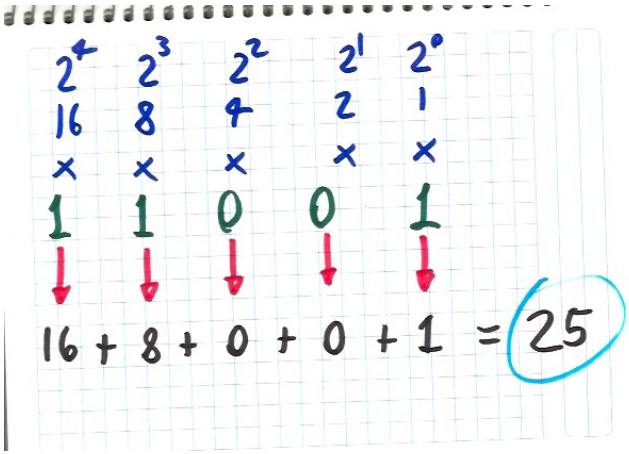
- **Sistemas de numeração:**
  - Decimal (Base 10: 0,1,2,3,4,5,6,7,8,9);
    $1998 = 1*10^3+9*10^2+9*10^1+8*10^0$
  - Binário (Base 2: 0,1);
  - Octal (Base 8: 0,1,2,3,4,5,6,7);
  - Hexadecimal (Base 16: 0,..9,A,B,C,D,E,F)

- Bit – unidade mínima de informação dos sistemas digitais;
  - **B**inary Dig**it** (BIT) (0 1) (2^1)
- Byte - 8 bits -> ($256 = 2^8$)

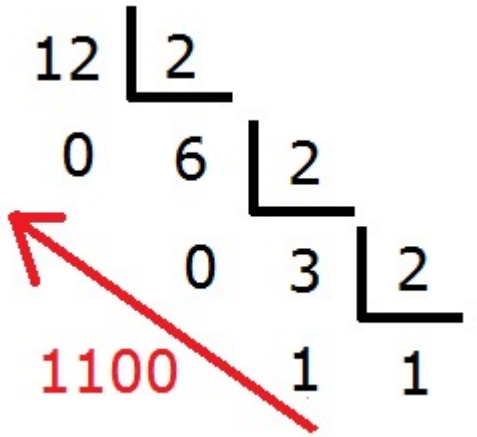
- 1 Byte = 8 bits
- 1 Kbyte = 1024 bytes
- 1 Mbyte = 1024 Kbytes
- 1 Gbyte = 1024 Mbytes
- 1 Tbyte = 1024 Gbytes

Sistemas de numeração

- **Conversão de binária/decimal:**



Conversão Binária->Decimal



Conversão Decimal->Binária

Bibliografia

Learn to Program with Python 3 - A Step-by-Step Guide to Programming -Second Edition

https://www.zerynth.com/zsdk/ [IDE-Zerinth]

https://www.open-electronics.org/python-on-esp32-easy-for-beginners-powerful-for-professionals/

https://thonny.org

https://www.w3schools.com/python/